EAST Search History

Ref #	Hits	Search Query	DBs	Default Operator	Plurals	Time Stamp
L2	7	(fill$4 same spill$4) same (rotat$4 adj register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:10
L3	7	(fill$4 and spill$4) same (rotat$4 adj register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 15:30
L5	24	(fill$4 and spill$4) and (rotat$4 adj register) and stack	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:10
L6	32	(fill$4 and spill$4) and (rotat$4 adj register) and (float$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:11
L7	1	(fill$4 and spill$4) same (rotat$4 adj register) same (float$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:11
L9	2	(fill$4 and spill$4) same (rotat$4 adj register) same (memory)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 16:01
L13	4	(fill$4 spill$4) same (rotat$4 adj register) same stack	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:06
L14	8	(fill$4 spill$4) same (rotat$4 adj register) same compil$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:07

EAST Search History

L15	14	(float$4 near4 rotating near7 register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:12
L16	3	(float$4 near4 rotating near7 register) same (spill$4 fill$4)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:12
L20	4	(float$4 near4 rotating near7 register) and compil$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 16:40
L25	53	(spill$4 load$4 stor$4) same (rotat$4 adj register) and compil$4	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 16:32
L27	33	(software near3 loop near3 pipelin4) same (rotat$4 adj register)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:12
L29	37	(software near3 loop near3 pipelin4) same (epolog prolog)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 16:50
L30	8	(fill$4 spill$4) same (rotat$4 adj register) and (717/150 717/151 717/160 717/161)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:01
L32	6	(fill$4 and spill$4) and (rotat$4 adj register) and (717/150 717/151 717/160 717/161)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:03

EAST Search History

L33	3	(fill$4 and spill$4) and (rotat$4 adj register) and stack and (717/150 717/151 717/160 717/161)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:04
L37	4	(fill$4 spill$4) same (rotat$4 adj register) same compil$4 and (717/150 717/151 717/160 717/161)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:07
L40	3	(float$4 near4 rotating near7 register) and (717/150 717/151 717/160 717/161 "712"/$.ccls.)	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:08
L42	3	(fill$4 spill$4) same (rotat$4 adj register).clm.	US-PGPUB	OR	ON	2007/07/24 17:10
L43	2	(fill$4 same spill$4) same (rotat$4 adj register).clm.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:10
L44	7	(fill$4 and spill$4) and (rotat$4 adj register).clm.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:10
L45	4	(fill$4 and spill$4) and (rotat$4 adj register) and stack.clm.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:11
L46	11	(fill$4 and spill$4) and (rotat$4 adj register) and (float$4).clm.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:11
L48	3	(float$4 near4 rotating near7 register).clm.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:12

EAST Search History

L50	3	(software near3 loop near3 pipelin4) same (rotat$4 adj register).clm.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:12
L51	45	(software near3 loop near3 pipelin4).clm.	US-PGPUB; USPAT; USOCR; EPO; JPO; DERWENT; IBM_TDB	OR	ON	2007/07/24 17:13

PORTAL USPTO
Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
+rotat* and register and spill* SEARCH
THE ACM DIGITAL LIBRARY
Feedback Report a problem Satisfaction survey

Terms used: **rotat** and **register** and **spill** Found **2,542** of **207,474**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window
Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Embedded software: task scheduling and compilation: Register aware scheduling for distributed cache clustered architecture
Zhong Wang, Xiaobo Sharon Hu, Edwin H.-M. Sha
January 2003 **Proceedings of the 2003 conference on Asia South Pacific design automation ASPDAC**
Publisher: ACM Press
Full text available: pdf(117.33 KB) Additional Information: full citation, abstract, references

Increasing wire delays have become a serious problem for sophisticated VLSI designs. Clustered architecture offers a promising alternative to alleviate the problem. In the clustered architecture, the cache, register file and function units are all partitioned into clusters such that short CPU cycle time can be achieved. A key challenge is the arrangement of inter-cluster communication. In this paper, we present a novel algorithm for scheduling inter-cluster communication operations. Our algorith ...

2 Heuristics for register-constrained software pipelining
Josep Llosa, Mateo Valero, Eduard Ayguadé
December 1996 **Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture MICRO 29**
Publisher: IEEE Computer Society
Full text available: pdf(1.44 MB) Additional Information: full citation, abstract, references, citings, index terms

Software Pipelining is a loop scheduling technique that extracts parallelism from loops by overlapping the execution of several consecutive iterations. There has been a significant effort to produce throughput-optimal schedules under resource constraints, and more recently to produce throughput-optimal schedules with minimum register requirements. Unfortunately even a throughput-optimal schedule with minimum register requirements is useless if it requires more registers than those available in t ...

3 Dependences and register allocation: Prematerialization: reducing register pressure for free
Ivan D. Baev, Richard E. Hank, David H. Gross
September 2006 **Proceedings of the 15th international conference on Parallel architectures and compilation techniques PACT '06**
Publisher: ACM Press
Full text available: pdf(159.54 KB) Additional Information: full citation, abstract, references, index terms

Modern compiler transformations that eliminate redundant computations or reorder instructions, such as partial redundancy elimination and instruction scheduling, are very effective in improving application performance but tend to create longer and potentially more complex live ranges. Typically the task of dealing with the increased register pressure is left to the register allocator. To avoid introduction of spill code which can

reduce or completely eliminate the benefit of earlier optimization ...

Keywords: Itanium, VLIW, register allocation, register pressure, rematerialization

4 Parallelization of loops with exits on pipelined architectures

P. Tirumalai, M. Lee, M. Schlansker

November 1990 **Proceedings of the 1990 ACM/IEEE conference on Supercomputing Supercomputing '90**

Publisher: IEEE Computer Society

Full text available: pdf(1.20 MB) Additional Information: full citation, abstract, references, citings

Modulo scheduling theory can be applied successfully to overlap Fortran DO loops on pipelined computers issuing multiple operations per cycle both with and without special loop architectural support [l, 2, 3]. This paper shows that a broader class of loops - *repeat-until, while,* and loops with more than one exit - where the trip count is not known beforehand, can also be overlapped efficiently on multiple issue pipelined machines. Special features that are required in the architecture, as ...

Keywords: Conditional execution, dependence graphs, loop scheduling, modulo scheduling, performance bounds, pipelined architectures, software pipelining, while loops

5 Code scheduling: Phi-Predication for light-weight if-conversion

Weihaw Chuang, Brad Calder, Jeanne Ferrante

March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**

Publisher: IEEE Computer Society

Full text available: pdf(1.19 MB) Additional Information: full citation, abstract, references, citings, index terms

Predicated execution can eliminate hard to predict branches and help to enable instruction level parallelism. Many current predication variants exist where the result update is conditional based upon the outcome of the guarding predicate. However, conditional writing of a register creates a naming problem for an out-of-order processor, and can stall the issuing of instructions. This problem arises from potential multiple predicated definitions reaching a use, which is unresolved until the prior ...

6 Allocating architected registers through differential encoding

Xiaotong Zhuang, Santosh Pande

April 2007 **ACM Transactions on Programming Languages and Systems (TOPLAS)**, Volume 29 Issue 2

Publisher: ACM Press

Full text available: pdf(312.02 KB) Additional Information: full citation, abstract, references, index terms

Micro-architecture designers are very cautious about expanding the number of architected and exposed registers in the instruction set because increasing the register field adds to the code size, raises the I-cache and memory pressure, and may complicate the processor pipeline. Especially for low-end processors, encoding space could be extremely limited due to area and power considerations. On the other hand, the number of architected registers exposed to the compiler could directly affect the ...

Keywords: Register allocation, architected register, differential encoding

7 Differential register allocation

Xiaotong Zhuang, Santosh Pande

June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40 Issue 6

Publisher: ACM Press

Full text available: pdf(214.10 KB) Additional Information: full citation, abstract, references, index terms

Micro-architecture designers are very cautious about expanding the number of architected registers (also the register field), because increasing the register field adds to the code size, raises I-cache and memory pressure, complicates processor pipeline. Especially for low-end processors, encoding space could be extremely limited due to area and power considerations. On the other hand, the number of architected registers exposed to the compiler could directly affect the effectiveness of compiler ...

Keywords: architected register, differential dncoding, register allocation

8 Processor microarchitecture II: Predicate prediction for efficient out-of-order execution

Weihaw Chuang, Brad Calder

June 2003 **Proceedings of the 17th annual international conference on Supercomputing ICS '03**

Publisher: ACM Press

Full text available: pdf(220.52 KB) Additional Information: full citation, abstract, references, index terms

Predicated execution is an important optimization even for an out-of-order processor, since it can eliminate hard to predict branches and help to enable software pipelining. Using predication with out-of-order execution creates a naming bottleneck, because there can be multiple definitions reaching a use, and not knowing which use is the correct one can stall the processor.In this paper, we examine using predicate prediction to speculatively allow execution to proceed in the face of multiple def ...

Keywords: predicate prediction, predicated execution

9 Demystifying on-the-fly spill code

Alex Aletà, Josep M. Codina, Antonio Gonzàlez, David Kaeli

June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40 Issue 6

Publisher: ACM Press

Full text available: pdf(192.08 KB) Additional Information: full citation, abstract, references, index terms

Modulo scheduling is an effective code generation technique that exploits the parallelism in program loops by overlapping iterations. One drawback of this optimization is that register requirements increase significantly because values across different loop iterations can be live concurrently. One possible solution to reduce register pressure is to insert spill code to release registers. Spill code stores values to memory between the producer and consumer instructions.Spilling heuristics can be ...

Keywords: modulo scheduling, register allocation, spill code

10 A performance analysis of PIM, stream processing, and tiled processing on memory-intensive signal processing kernels

Jinwoo Suh, Eun-Gyu Kim, Stephen P. Crago, Lakshmi Srinivasan, Matthew C. French

May 2003 **ACM SIGARCH Computer Architecture News , Proceedings of the 30th annual international symposium on Computer architecture ISCA '03**, Volume 31 Issue 2

Publisher: ACM Press

Full text available: pdf(239.50 KB) Additional Information: full citation, abstract, references, citings

Trends in microprocessors of increasing die size and clock speed and decreasing feature sizes have fueled rapidly increasing performance. However, the limited improvements in DRAM latency and bandwidth and diminishing returns of increasing superscalar ILP and cache sizes have led to the proposal of new microprocessor architectures that implement processor-in- memory, stream processing, and tiled processing. Each architecture is

typically evaluated separately and compared to a baseline architectu ...

11 RISCY patents

David A. Patterson
September 1988 **ACM SIGARCH Computer Architecture News**, Volume 16 Issue 4
Publisher: ACM Press
Full text available: pdf(1.83 MB) Additional Information: full citation, index terms

12 WBIA'05: Using Pin as a memory reference generator for multiprocessor simulation

Collin McCurdy, Charles Fischer
December 2005 **ACM SIGARCH Computer Architecture News**, Volume 33 Issue 5
Publisher: ACM Press
Full text available: pdf(249.21 KB) Additional Information: full citation, abstract, references, index terms

In this paper we describe how we have used Pin to generate a multithreaded reference stream for simulation of a multiprocessor on a uniprocessor. We have taken special care to model as accurately as possible the effects of cache coherence protocol state, and lock and barrier synchronization on the performance of multithreaded applications running on multiprocessor hardware.We first describe a simplified version of the algorithm, which uses semaphores to synchronize instrumented application threa ...

13 Workload optimization: Implicit and explicit optimizations for stencil computations

Shoaib Kamil, Kaushik Datta, Samuel Williams, Leonid Oliker, John Shalf, Katherine Yelick
October 2006 **Proceedings of the 2006 workshop on Memory system performance and correctness MSPC '06**
Publisher: ACM Press
Full text available: pdf(563.94 KB) Additional Information: full citation, abstract, references, index terms

Stencil-based kernels constitute the core of many scientific applications on block-structured grids. Unfortunately, these codes achieve a low fraction of peak performance, due primarily to the disparity between processor and main memory speeds. We examine several optimizations on both the conventional cache-based memory systems of the Itanium 2, Opteron, and Power5, as well as the heterogeneous multicore design of the Cell processor. The optimizations target cache reuse across stencil sweeps, in ...

14 Improved spill code generation for software pipelined loops

Javier Zalamea, Josep Llosa, Eduard Ayguadé, Mateo Valero
May 2000 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 2000 conference on Programming language design and implementation PLDI '00**, Volume 35 Issue 5
Publisher: ACM Press
Full text available: pdf(688.06 KB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining is a loop scheduling technique that extracts parallelism out of loops by overlapping the execution of several consecutive iterations. Due to the overlapping of iterations, schedules impose high register requirements during their execution. A schedule is valid if it requires at most the number of registers available in the target architecture. If not, its register requirements have to be reduced either by decreasing the iteration overlapping or by spilling registers t ...

Keywords: instruction-level parallelism, register allocation, software pipelining, spill code

15 Compilers II: Inter-procedural stacked register allocation for itanium® like architecture

Liu Yang, Sun Chan, G. R. Gao, Roy Ju, Guei-Yuan Lueh, Zhaoqing Zhang
June 2003 **Proceedings of the 17th annual international conference on**

Supercomputing ICS '03
Publisher: ACM Press

Full text available: pdf(478.20 KB) Additional Information: full citation, abstract, references, citings, index terms

A hardware managed register stack, Register Stack Engine (RSE), is implemented in Itanium® architecture to provide a unified and flexible register structure to software. The compiler allocates each procedure a register stack frame with its size explicitly specified using an *alloc* instruction. When the total number of registers used by the procedures on the call stack exceeds the number of physical registers, RSE performs automatically register overflows and fills to ensure that the c ...

Keywords: hot region, hotspot, inter-procedural stacked register allocation, quota assignment, register allocation

16 Register allocation for software pipelined loops

B. R. Rau, M. Lee, P. P. Tirumalai, M. S. Schlansker
July 1992 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1992 conference on Programming language design and implementation PLDI '92**, Volume 27 Issue 7
Publisher: ACM Press

Full text available: pdf(1.84 MB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining is an important instruction scheduling technique for efficiently overlapping successive iterations of loops and executing them in parallel. This paper studies the task of register allocation for software pipelined loops, both with and without hardware features that are specifically aimed at supporting software pipelines. Register allocation for software pipelines presents certain novel problems leading to unconventional solutions, especially in the presence of hardware s ...

17 Accelerator: using data parallelism to program GPUs for general-purpose uses

David Tarditi, Sidd Puri, Jose Oglesby
October 2006 **ACM SIGARCH Computer Architecture News , ACM SIGOPS Operating Systems Review , ACM SIGPLAN Notices , Proceedings of the 12th international conference on Architectural support for programming languages and operating systems ASPLOS-XII**, Volume 34 , 40 , 41 Issue 5 , 5 , 11
Publisher: ACM Press

Full text available: pdf(266.52 KB) Additional Information: full citation, abstract, references, index terms

GPUs are difficult to program for general-purpose uses. Programmers can either learn graphics APIs and convert their applications to use graphics pipeline operations or they can use stream programming abstractions of GPUs. We describe Accelerator, a system that uses data parallelism to program GPUs for general-purpose uses instead. Programmers use a conventional imperative programming language and a library that provides only high-level data-parallel operations. No aspects of GPUs are exposed to ...

Keywords: data parallelism, graphics processing units, just-in time compilation

18 Resource constrained dataflow retiming heuristics for VLIW ASIPs

M. Jacome, G. de Veciana, C. Akturan
March 1999 **Proceedings of the seventh international workshop on Hardware/software codesign CODES '99**
Publisher: ACM Press

Full text available: pdf(502.71 KB) Additional Information: full citation, references, citings, index terms

19 EPIC compilation: Optimization for the Intel® Itanium® architecture register stack

Alex Settle, Daniel A. Connors, Gerolf Hoflehner, Dan Lavery
March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**
Publisher: IEEE Computer Society

Full text available: pdf(906.60 KB) Additional Information: full citation, abstract, references, citings, index terms

The Intel® Itanium® architecture contains a number of innovative compiler-controllable features designed to exploit instruction level parallelism. New code generation and optimization techniques are critical to the application of these features to improve processor performance. For instance, the Itanium® architecture provides a compiler-controllable virtual register stack to reduce the penalty of memory accesses associated with procedure calls. The Itanium® Register Stack Engine ...

20 Overlapped loop support in the Cydra 5

James C. Dehnert, Peter Y.-T. Hsu, Joseph P. Bratt
April 1989 **ACM SIGARCH Computer Architecture News , Proceedings of the third international conference on Architectural support for programming languages and operating systems ASPLOS-III**, Volume 17 Issue 2
Publisher: ACM Press

Full text available: pdf(1.32 MB) Additional Information: full citation, abstract, references, citings, index terms

The CydraTM 5 architecture adds unique support for overlapping successive iterations of a loop to a very long instruction word (VLIW) base. This architecture allows highly parallel loop execution for a much larger class of loops than can be vectorized, without requiring the unrolling of loops usually used by compilers for VLIW machines. This paper discusses the Cydra 5 loop scheduling model, the special architectural features which support it, and the I ...

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL USPTO
Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
+rotat* and register and schedul* SEARCH
THE ACM DIGITAL LIBRARY
Feedback Report a problem Satisfaction survey

Terms used: **rotat** and **register** and **schedul** Found **4,296** of **207,474**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window

Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Embedded software: task scheduling and compilation: Register aware scheduling for distributed cache clustered architecture
Zhong Wang, Xiaobo Sharon Hu, Edwin H.-M. Sha
January 2003 **Proceedings of the 2003 conference on Asia South Pacific design automation ASPDAC**
Publisher: ACM Press
Full text available: pdf(117.33 KB) Additional Information: full citation, abstract, references

Increasing wire delays have become a serious problem for sophisticated VLSI designs. Clustered architecture offers a promising alternative to alleviate the problem. In the clustered architecture, the cache, register file and function units are all partitioned into clusters such that short CPU cycle time can be achieved. A key challenge is the arrangement of inter-cluster communication. In this paper, we present a novel algorithm for scheduling inter-cluster communication operations. Our algorith ...

2 Code generation schema for modulo scheduled loops
B. Ramakrishna Rau, Michael S. Schlansker, P. P. Tirumalai
December 1992 **ACM SIGMICRO Newsletter , Proceedings of the 25th annual international symposium on Microarchitecture MICRO 25**, Volume 23 Issue 1-2
Publisher: IEEE Computer Society Press, ACM Press
Full text available: pdf(1.49 MB) Additional Information: full citation, references, citings, index terms

3 A register file and scheduling model for application specific processor synthesis
E. Ercanli, C. Papachristou
June 1996 **Proceedings of the 33rd annual conference on Design automation DAC '96**
Publisher: ACM Press
Full text available: pdf(123.46 KB) Additional Information: full citation, references, citings, index terms

4 Register allocation for software pipelined loops
B. R. Rau, M. Lee, P. P. Tirumalai, M. S. Schlansker
July 1992 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1992 conference on Programming language design and implementation PLDI '92**, Volume 27 Issue 7
Publisher: ACM Press
Full text available: pdf(1.84 MB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining is an important instruction scheduling technique for efficiently

overlapping successive iterations of loops and executing them in parallel. This paper studies the task of register allocation for software pipelined loops, both with and without hardware features that are specifically aimed at supporting software pipelines. Register allocation for software pipelines presents certain novel problems leading to unconventional solutions, especially in the presence of hardware s ...

5 Lifetime-sensitive modulo scheduling

Richard A. Huff
June 1993 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1993 conference on Programming language design and implementation PLDI '93**, Volume 28 Issue 6
Publisher: ACM Press

Full text available: pdf(1.08 MB) Additional Information: full citation, abstract, references, citings, index terms, review

This paper shows how to software pipeline a loop for minimal register pressure without sacrificing the loop's minimum execution time. This novel bidirectional slack-scheduling method has been implemented in a FORTRAN compiler and tested on many scientific benchmarks. The empirical results—when measured against an absolute lower bound on execution time, and against a novel schedule-independent absolute lower bound on register pressure—indicate ne ...

6 Modulo scheduling with multiple initiation intervals

Nancy J. Warter-Perez, Noubar Partamian
December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture MICRO 28**
Publisher: IEEE Computer Society Press
Full text available: pdf(885.83 KB) Additional Information: full citation, references, citings, index terms

7 Minimum register requirements for a modulo schedule

Alexandre E. Eichenberger, Edward S. Davidson, Santosh G. Abraham
November 1994 **Proceedings of the 27th annual international symposium on Microarchitecture MICRO 27**
Publisher: ACM Press

Full text available: pdf(999.10 KB) Additional Information: full citation, abstract, references, citings, index terms

Modulo scheduling is an efficient technique for exploiting instruction level parallelism in a variety of loops, resulting in high performance code but increased register requirements. We present a combined approach that schedules the loop operations for minimum register requirements, given a modulo reservation table. Our method determines optimal register requirements for machines with finite resources and for general dependence graphs. This method demonstrates the potential of lifetime-sen ...

8 Optimum modulo schedules for minimum register requirements

Alexandre E. Eichenberger, Edward S. Davidson, Santosh G. Abraham
July 1995 **Proceedings of the 9th international conference on Supercomputing ICS '95**
Publisher: ACM Press
Full text available: pdf(1.20 MB) Additional Information: full citation, references, citings, index terms

9 Modulo scheduling: Modulo schedule buffers

Matthew C. Merten, Wen-mei W. Hwu
December 2001 **Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture MICRO 34**
Publisher: IEEE Computer Society
Full text available:

pdf(1.38 MB) Additional Information: full citation, abstract, references, citings
Publisher Site

As VLIW/EPIC processors are increasingly used in real-time, signal-processing, and embedded applications, the importance of minimizing code size and reducing power is growing. This paper describes a new architectural mechanism, called the Modulo Schedule Buffers, that provides an elegant interface for the execution of modulo scheduled loops. While the performance is similar to that of kernel-only modulo scheduling, this mechanism has a number of advantages, including minimal code expansion. Rath ...

10 Compiler scheduling: Reduced code size modulo scheduling in the absence of hardware support

Josep Llosa, Stefan M. Freudenberger
November 2002 **Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture MICRO 35**
Publisher: IEEE Computer Society Press
Full text available: pdf(1.17 MB) Additional Information: full citation, abstract, references, index terms
Publisher Site

Modulo scheduling is a very effective instruction scheduling technique that exploits Instruction Level Parallelism (ILP) in loop bodies by overlapping the execution of successive iterations. Unfortunately, modulo scheduling has been shown to cause heavy code expansion. To avoid the penalties of code expansion, some processors have dedicated hardware support for modulo scheduled loops. However, this dedicated hardware support has a cost in chip area, cycle time, processor complexity, and compiler ...

11 Register allocation for software pipelined multi-dimensional loops

Hongbo Rong, Alban Douillet, Guang R. Gao
June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40 Issue 6
Publisher: ACM Press
Full text available: pdf(745.17 KB) Additional Information: full citation, abstract, references, index terms

Software pipelining of a multi-dimensional loop is an important optimization that overlaps the execution of successive outermost loop iterations to explore instruction-level parallelism from the entire n-dimensional iteration space. This paper investigates register allocation for software pipelined multi-dimensional loops.For single loop software pipelining, the lifetime instances of a loop variant in successive iterations of the loop form a repetitive pattern. An effective register allocation m ...

Keywords: register allocation, software pipelining

12 RS-FDRA: a register sensitive software pipelining algorithm for embedded VLIW processors

Cagdas Akturan, Margarida F. Jacome
April 2001 **Proceedings of the ninth international symposium on Hardware/software codesign CODES '01**
Publisher: ACM Press
Full text available: pdf(848.80 KB) Additional Information: full citation, abstract, references, citings, index terms

The paper proposes a novel software-pipelining algorithm, *Register Sensitive Force Directed Retiming Algorithm (RS-FDRA)*, suitable for optimizing compilers targeting embedded VLIW processors. The key difference between RS-FDRA and previous approaches is that our algorithm can handle *code size constraints* along with latency and resource constraints. This capability enables the exploration of pareto "optimal" points with respect to *code size* and *performance*
***Keywords**: VLIW processors, embedded systems, optimizing compilers, retiming,*

software pipelining

13 An efficient cell-scheduling algorithm for multicast ATM switching systems
Wen-Tsuen Chen, Chun-Fu Huang, Yi-Luang Chang, Wu-Yuin Hwang
August 2000 **IEEE/ACM Transactions on Networking (TON)**, Volume 8 Issue 4
Publisher: IEEE Press
Full text available: pdf(254.39 KB) Additional Information: full citation, references, citings, index terms

Keywords: asynchronous transfer mode, cell-scheduling algorithm, head-of-line blocking problem, multicast

14 Stage scheduling: a technique to reduce the register requirements of a modulo schedule
Alexandre E. Eichenberger, Edward S. Davidson
December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture MICRO 28**
Publisher: IEEE Computer Society Press
Full text available: pdf(1.24 MB) Additional Information: full citation, references, citings, index terms

15 Code size reduction technique and implementation for software-pipelined DSP applications
Qingfeng Zhuge, Bin Xiao, Edwin H.-M. Sha
November 2003 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 2 Issue 4
Publisher: ACM Press
Full text available: pdf(371.88 KB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining technique is extensively used to exploit instruction-level parallelism of loops, but also significantly expands the code size. For embedded systems with very limited on-chip memory resources, code size becomes one of the most important optimization concerns. This paper presents the theoretical foundation of code size reduction for software-pipelined loops based on retiming concept. We propose a general Code-size REDuction technique (CRED) for various kinds of processors. Our ...

Keywords: DSP processors, Retiming, scheduling, software pipelining

16 Probabilistic Predicate-Aware Modulo Scheduling
Mikhail Smelyanskiy, Scott Mahlke, Edward S. Davidson
March 2004 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '04**
Publisher: IEEE Computer Society
Full text available: pdf(365.00 KB) Additional Information: full citation, abstract, index terms

Predicated execution enables the removal of branches by convertingsegments of branching code into sequences of conditional operations.An important side effect of this transformation is that thecompiler must unconditionally assign resources to predicated operations.However, a resource is only put to productive use whenthe predicate associated with an operation evaluates to True. To reducethis superfluous commitment of resources, we propose probabilisticpredicate-aware scheduling to assign multipl ...

17 Software pipelining
Vicki H. Allan, Reese B. Jones, Randall M. Lee, Stephen J. Allan

September 1995 **ACM Computing Surveys (CSUR)**, Volume 27 Issue 3
Publisher: ACM Press
Full text available: pdf(4.72 MB) Additional Information: full citation, abstract, references, citings, index terms

Utilizing parallelism at the instruction level is an important way to improve performance. Because the time spent in loop execution dominates total execution time, a large body of optimizations focuses on decreasing the time to execute each iteration. Software pipelining is a technique that reforms the loop so that a faster execution rate is realized. Iterations are executed in overlapped fashion to increase parallelism.Let {ABC}n
Keywords: instruction level parallelism, loop reconstruction, optimization, software pipelining

18 Session 7B: Architecture oriented scheduling: Constraint satisfaction for relative location assignment and scheduling

Carlos Alba-Pinto, Bart Mesman, Jochen Jess
November 2001 **Proceedings of the 2001 IEEE/ACM international conference on Computer-aided design ICCAD '01**
Publisher: IEEE Press
Full text available: pdf(129.20 KB) Additional Information: full citation, abstract, references, index terms

Tight data-and timing constraints are imposed by communication and multimedia applications. The architecture for the embedded processor imply resource constraints. Instead of random-access registers, relative location storages or rotating register files are used to exploit the available parallelism of resources by means of reducing the initiation interval in pipelined schedules. Therefore, the compiler or synthesis tool must deal with the difficult tasks of scheduling of operations and location ...

19 Effective cluster assignment for modulo scheduling

Erik Nystrom, Alexandre E. Eichenberger
November 1998 **Proceedings of the 31st annual ACM/IEEE international symposium on Microarchitecture MICRO 31**
Publisher: IEEE Computer Society Press
Full text available: pdf(1.72 MB) Additional Information: full citation, references, citings, index terms

Keywords: ILP, cluster architecture, cluster assignment, modulo scheduling

20 Iterative modulo scheduling: an algorithm for software pipelining loops

B. Ramakrishna Rau
November 1994 **Proceedings of the 27th annual international symposium on Microarchitecture MICRO 27**
Publisher: ACM Press
Full text available: pdf(1.49 MB) Additional Information: full citation, abstract, references, citings, index terms

Modulo scheduling is a framework within which a wide variety of algorithms and heuristics may be defined for software pipelining innermost loops. This paper presents a practical algorithm, iterative modulo scheduling, that is capable of dealing with realistic machine models. This paper also characterizes the algorithm in terms of the quality of the generated schedules as well the computational expense incurred.

Keywords: instruction scheduling, loop scheduling, modulo scheduling, software pipelining

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL USPTO
Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
+rotat* and register and pipelin* SEARCH
THE ACM DIGITAL LIBRARY
Feedback Report a problem Satisfaction survey

Terms used: **rotat** and **register** and **pipelin** Found **3,311** of **207,474**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window
Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Register allocation for software pipelined loops
B. R. Rau, M. Lee, P. P. Tirumalai, M. S. Schlansker
July 1992 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1992 conference on Programming language design and implementation PLDI '92**, Volume 27 Issue 7
Publisher: ACM Press
Full text available: pdf(1.84 MB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining is an important instruction scheduling technique for efficiently overlapping successive iterations of loops and executing them in parallel. This paper studies the task of register allocation for software pipelined loops, both with and without hardware features that are specifically aimed at supporting software pipelines. Register allocation for software pipelines presents certain novel problems leading to unconventional solutions, especially in the presence of hardware s ...

2 Register allocation for software pipelined multi-dimensional loops
Hongbo Rong, Alban Douillet, Guang R. Gao
June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40 Issue 6
Publisher: ACM Press
Full text available: pdf(745.17 KB) Additional Information: full citation, abstract, references, index terms

Software pipelining of a multi-dimensional loop is an important optimization that overlaps the execution of successive outermost loop iterations to explore instruction-level parallelism from the entire n-dimensional iteration space. This paper investigates register allocation for software pipelined multi-dimensional loops.For single loop software pipelining, the lifetime instances of a loop variant in successive iterations of the loop form a repetitive pattern. An effective register allocation m ...

Keywords: register allocation, software pipelining

3 Software pipelining
Vicki H. Allan, Reese B. Jones, Randall M. Lee, Stephen J. Allan
September 1995 **ACM Computing Surveys (CSUR)**, Volume 27 Issue 3
Publisher: ACM Press
Full text available: pdf(4.72 MB) Additional Information: full citation, abstract, references, citings, index terms

Utilizing parallelism at the instruction level is an important way to improve performance. Because the time spent in loop execution dominates total execution time, a large body of

optimizations focuses on decreasing the time to execute each iteration. Software pipelining is a technique that reforms the loop so that a faster execution rate is realized. Iterations are executed in overlapped fashion to increase parallelism.Let {ABC}n
Keywords: instruction level parallelism, loop reconstruction, optimization, software pipelining

4 Code Generation for Single-Dimension Software Pipelining of Multi-Dimensional Loops

Hongbo Rong, Alban Douillet, R. Govindarajan, Guang R. Gao
March 2004 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '04**
Publisher: IEEE Computer Society
Full text available: pdf(594.36 KB) Additional Information: full citation, abstract, citings, index terms

Traditionally, software pipelining is applied either to theinnermost loop of a given loop nest or from the innermostloop to the outer loops. In a companion paper, we proposeda scheduling method, called Single-dimension SoftwarePipelining (SSP), to software pipeline a multi-dimensionalloop nest at an arbitrary loop level.In this paper, we describe our solution to SSP code generation.In contrast to traditional software pipelining, SSPhandles two distinct repetitive patterns, and thus requiresnew c ...

5 Code size reduction technique and implementation for software-pipelined DSP applications

Qingfeng Zhuge, Bin Xiao, Edwin H.-M. Sha
November 2003 **ACM Transactions on Embedded Computing Systems (TECS)**, Volume 2 Issue 4
Publisher: ACM Press
Full text available: pdf(371.88 KB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining technique is extensively used to exploit instruction-level parallelism of loops, but also significantly expands the code size. For embedded systems with very limited on-chip memory resources, code size becomes one of the most important optimization concerns. This paper presents the theoretical foundation of code size reduction for software-pipelined loops based on retiming concept. We propose a general Code-size REDuction technique (CRED) for various kinds of processors. Our ...

Keywords: DSP processors, Retiming, scheduling, software pipelining

6 Single-dimension software pipelining for multidimensional loops

Hongbo Rong, Zhizhong Tang, R. Govindarajan, Alban Douillet, Guang R. Gao
March 2007 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 4 Issue 1
Publisher: ACM Press
Full text available: pdf(439.64 KB) Additional Information: full citation, abstract, references, index terms

Traditionally, software pipelining is applied either to the innermost loop of a given loop nest or from the innermost loop to outer loops. This paper proposes a three-step approach, called *single-dimension software pipelining (SSP)*, to software pipeline a loop nest at an arbitrary loop level that has a rectangular iteration space and contains no sibling inner loops in it. The first step identifies the most profitable loop level for software pipelining in terms of initiation rate, data ...

Keywords: Software pipelining, loop transformation, modulo scheduling

7 Pseudo vector processor based on register-windowed superscalar pipeline

K. Nakazawa, H. Nakamura, H. Imori, S. Kawabe
December 1992 **Proceedings of the 1992 ACM/IEEE conference on Supercomputing**

Supercomputing '92
Publisher: IEEE Computer Society Press
Full text available: pdf(1.19 MB) Additional Information: full citation, references, citings, index terms

8 Embedded software: task scheduling and compilation: Register aware scheduling for distributed cache clustered architecture
Zhong Wang, Xiaobo Sharon Hu, Edwin H.-M. Sha
January 2003 **Proceedings of the 2003 conference on Asia South Pacific design automation ASPDAC**
Publisher: ACM Press
Full text available: pdf(117.33 KB) Additional Information: full citation, abstract, references

Increasing wire delays have become a serious problem for sophisticated VLSI designs. Clustered architecture offers a promising alternative to alleviate the problem. In the clustered architecture, the cache, register file and function units are all partitioned into clusters such that short CPU cycle time can be achieved. A key challenge is the arrangement of inter-cluster communication. In this paper, we present a novel algorithm for scheduling inter-cluster communication operations. Our algorith ...

9 RS-FDRA: a register sensitive software pipelining algorithm for embedded VLIW processors
Cagdas Akturan, Margarida F. Jacome
April 2001 **Proceedings of the ninth international symposium on Hardware/software codesign CODES '01**
Publisher: ACM Press
Full text available: pdf(848.80 KB) Additional Information: full citation, abstract, references, citings, index terms

The paper proposes a novel software-pipelining algorithm, *Register Sensitive Force Directed Retiming Algorithm (RS-FDRA)*, suitable for optimizing compilers targeting embedded VLIW processors. The key difference between RS-FDRA and previous approaches is that our algorithm can handle *code size constraints* along with latency and resource constraints. This capability enables the exploration of pareto "optimal" points with respect to *code size* and *performance*
***Keywords**: VLIW processors, embedded systems, optimizing compilers, retiming, software pipelining*

10 Software Pipelining Irregular Loops On the TMS320C6000 VLIW DSP Architecture
Elana Granston, Eric Stotzer, Joe Zbiciak
August 2001 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN workshop on Languages, compilers and tools for embedded systems LCTES '01 , Proceedings of the 2001 ACM SIGPLAN workshop on Optimization of middleware and distributed systems OM '01**, Volume 36 Issue 8
Publisher: ACM Press
Full text available: pdf(88.54 KB) Additional Information: full citation, abstract, references, index terms

The TMS320C6000 architecture is a leading family of Digital Signal Processors (DSPs). To achieve peak performance, this VLIW architecture relies heavily on software pipelining. Traditionally, software pipelining has been restricted to *regular* (FOR) loops. More recently, software pipelining has been extended to *irregular* (WHILE) loops, but only on architectures that provide special-purpose hardware such as rotating (predicate and general-purpose) register files, specific instruct ...

Keywords: VLIW architectures, WHILE loops, digital signal processors, software pipelining

11 Session S9.1: software transformation: Validating software pipelining optimizations

Raya Leviathan, Amir Pnueli
October 2002 **Proceedings of the 2002 international conference on Compilers, architecture, and synthesis for embedded systems CASES '02**
Publisher: ACM Press

Full text available: pdf(208.42 KB) Additional Information: full citation, abstract, references, citings, index terms

The paper presents a method for translation validation of a specific optimization, software pipelining optimization, used to increase the instruction level parallelism in EPIC type of architectures. Using a methodology as in [15] to establish simulation relation between source and target based on computational induction, we describe an algorithm that automatically produces a set of decidable proof obligations. The paper also describes *SPV*, a prototype translation validator that automatica ...

Keywords: compilers, optimization, pipeline processors, translation validation, verification

12 Modulo scheduling with multiple initiation intervals
Nancy J. Warter-Perez, Noubar Partamian
December 1995 **Proceedings of the 28th annual international symposium on Microarchitecture MICRO 28**
Publisher: IEEE Computer Society Press
Full text available: pdf(885.83 KB) Additional Information: full citation, references, citings, index terms

13 A register file and scheduling model for application specific processor synthesis
E. Ercanli, C. Papachristou
June 1996 **Proceedings of the 33rd annual conference on Design automation DAC '96**
Publisher: ACM Press
Full text available: pdf(123.46 KB) Additional Information: full citation, references, citings, index terms

14 Scientific computing on the Itanium™ processor
Bruce Greer, John Harrison, Greg Henry, Wei Li, Peter Tang
November 2001 **Proceedings of the 2001 ACM/IEEE conference on Supercomputing (CDROM) Supercomputing '01**
Publisher: ACM Press
Full text available: pdf(177.31 KB) Additional Information: full citation, abstract, references, index terms

The 64-bit Intel® Itanium™ architecture is designed for high-performance scientific and enterprise computing, and the Itanium processor is its first silicon implementation. Features such as extensive arithmetic support, predication, speculation, and explicit parallelism can be used to provide a sound infrastructure for supercomputing. A large number of high-performance computer companies are offering Itanium™-based systems, some capable of peak performance exceeding 50 GFLOPS. In ...

Keywords: EPIC, fused multiply-add, itanium (TM) processor, linear algebra, transcendental functions

15 High-speed parameterisable Hough transform using reconfigurable hardware
Dixon D. S. Deng, Hossam ElGindy
May 2001 **Proceedings of the Pan-Sydney area workshop on Visual information processing - Volume 11 VIP '01**
Publisher: Australian Computer Society, Inc.
Full text available: pdf(82.01 KB) Additional Information: full citation, abstract, references, index terms

Recent developments in reconfigurable hardware technologies have offered high-density

high-speed devices with the ability for custom computing whilst maintaining the flexibility of a software solution. These features are well suited to image processing algorithms that are computationally intensive and repetitive in nature. Very deep pipelining and parallelism, features often required for real time image analysis can be achieved easily using hardware design. The Hough Transform is a powerful and ...

Keywords: CORDIC, Hough transform, reconfigurable hardware

16 Modulo scheduling: Modulo schedule buffers

Matthew C. Merten, Wen-mei W. Hwu

December 2001 **Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture MICRO 34**

Publisher: IEEE Computer Society

Full text available: pdf(1.38 MB) Additional Information: full citation, abstract, references, citings

Publisher Site

As VLIW/EPIC processors are increasingly used in real-time, signal-processing, and embedded applications, the importance of minimizing code size and reducing power is growing. This paper describes a new architectural mechanism, called the Modulo Schedule Buffers, that provides an elegant interface for the execution of modulo scheduled loops. While the performance is similar to that of kernel-only modulo scheduling, this mechanism has a number of advantages, including minimal code expansion. Rath ...

17 Dependences and register allocation: Prematerialization: reducing register pressure for free

Ivan D. Baev, Richard E. Hank, David H. Gross

September 2006 **Proceedings of the 15th international conference on Parallel architectures and compilation techniques PACT '06**

Publisher: ACM Press

Full text available: pdf(159.54 KB) Additional Information: full citation, abstract, references, index terms

Modern compiler transformations that eliminate redundant computations or reorder instructions, such as partial redundancy elimination and instruction scheduling, are very effective in improving application performance but tend to create longer and potentially more complex live ranges. Typically the task of dealing with the increased register pressure is left to the register allocator. To avoid introduction of spill code which can reduce or completely eliminate the benefit of earlier optimization ...

Keywords: Itanium, VLIW, register allocation, register pressure, rematerialization

18 Software pipelining loops with conditional branches

Mark G. Stoodley, Corinna G. Lee

December 1996 **Proceedings of the 29th annual ACM/IEEE international symposium on Microarchitecture MICRO 29**

Publisher: IEEE Computer Society

Full text available: pdf(1.64 MB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining is an aggressive scheduling technique that generates efficient code for loops and is particularly effective for VLIW architectures. Few software pipelining algorithms, however, are able to efficiently schedule loops that contain conditional branches. We have developed an algorithm we call All Paths Pipelining (APP) that addresses this shortcoming of software pipelining. APP is designed to achieve optimal or near-optimal performance for any run of iterations while providing ef ...

19 Code generation schema for modulo scheduled loops

B. Ramakrishna Rau, Michael S. Schlansker, P. P. Tirumalai

December 1992 **ACM SIGMICRO Newsletter , Proceedings of the 25th annual**

international symposium on Microarchitecture MICRO 25, Volume 23 Issue 1-2
Publisher: IEEE Computer Society Press, ACM Press
Full text available: pdf(1.49 MB) Additional Information: full citation, references, citings, index terms

20 Internet Nuggets
Mark Thorson
June 1999 **ACM SIGARCH Computer Architecture News**, Volume 27 Issue 3
Publisher: ACM Press
Full text available: pdf(591.26 KB) Additional Information: full citation, abstract

This column consists of selected traffic from the *comp.arch* newsgroup, a forum for discussion of computer architecture on Internet---an international computer network.

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

PORTAL USPTO
Subscribe (Full Service) Register (Limited Service, Free) Login
Search: The ACM Digital Library The Guide
+rotat* and register and fill* and compil* SEARCH
THE ACM DIGITAL LIBRARY
Feedback Report a problem Satisfaction survey

Terms used: **rotat** and **register** and **fill** and **compil** Found **6,032** of **207,474**

Sort results by: relevance
Display results: expanded form
Save results to a Binder
Search Tips
Open results in a new window
Try an Advanced Search
Try this search in The ACM Guide

Results 1 - 20 of 200 Result page: **1** 2 3 4 5 6 7 8 9 10 next
Best 200 shown Relevance scale

1 Attacking the semantic gap between application programming languages and configurable hardware
Greg Snider, Barry Shackleford, Richard J. Carter
February 2001 **Proceedings of the 2001 ACM/SIGDA ninth international symposium on Field programmable gate arrays FPGA '01**
Publisher: ACM Press
Full text available: pdf(258.65 KB) Additional Information: full citation, abstract, references, citings, index terms

It is difficult to exploit the massive, fine-grained parallelism of configurable hardware with a conventional application programming language such as C, Pascal or Java. The difficulty arises from the mismatch between the synchronous, concurrent processing capability of the hardware and the expressiveness of the language-the so-called "semantic gap." We attack this problem by using a programming model matched to the hardware's capabilities that can be implemented in any (unmodified) objec ...

2 Software Pipelining Irregular Loops On the TMS320C6000 VLIW DSP Architecture
Elana Granston, Eric Stotzer, Joe Zbiciak
August 2001 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN workshop on Languages, compilers and tools for embedded systems LCTES '01 , Proceedings of the 2001 ACM SIGPLAN workshop on Optimization of middleware and distributed systems OM '01**, Volume 36 Issue 8
Publisher: ACM Press
Full text available: pdf(88.54 KB) Additional Information: full citation, abstract, references, index terms

The TMS320C6000 architecture is a leading family of Digital Signal Processors (DSPs). To achieve peak performance, this VLIW architecture relies heavily on software pipelining. Traditionally, software pipelining has been restricted to *regular* (FOR) loops. More recently, software pipelining has been extended to *irregular* (WHILE) loops, but only on architectures that provide special-purpose hardware such as rotating (predicate and general-purpose) register files, specific instruct ...

Keywords: VLIW architectures, WHILE loops, digital signal processors, software pipelining

3 Compilation: The impact of if-conversion and branch prediction on program execution on the Intel® Itanium™ processor
Youngsoo Choi, Allan Knies, Luke Gerke, Tin-Fook Ngai
December 2001 **Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture MICRO 34**
Publisher: IEEE Computer Society

Full text available: pdf(1.06 MB) Additional Information: full citation, abstract, references, citings
Publisher Site

The research community has studied if-conversion for many years. However, due to the lack of existing hardware, studies were conducted by simulating code generated by experimental compilers. This paper presents the first comprehensive study of the use of predication to implement if-conversion on production hardware with a near-production compiler. To better understand trends in the measurements, we generated binaries at three increasing levels of if-conversion aggressiveness. For each level, we ...

4 Compilers II: Inter-procedural stacked register allocation for itanium® like architecture

Liu Yang, Sun Chan, G. R. Gao, Roy Ju, Guei-Yuan Lueh, Zhaoqing Zhang
June 2003 **Proceedings of the 17th annual international conference on Supercomputing ICS '03**
Publisher: ACM Press

Full text available: pdf(478.20 KB) Additional Information: full citation, abstract, references, citings, index terms

A hardware managed register stack, Register Stack Engine (RSE), is implemented in Itanium® architecture to provide a unified and flexible register structure to software. The compiler allocates each procedure a register stack frame with its size explicitly specified using an *alloc* instruction. When the total number of registers used by the procedures on the call stack exceeds the number of physical registers, RSE performs automatically register overflows and fills to ensure that the c ...

Keywords: hot region, hotspot, inter-procedural stacked register allocation, quota assignment, register allocation

5 Dependences and register allocation: Prematerialization: reducing register pressure for free

Ivan D. Baev, Richard E. Hank, David H. Gross
September 2006 **Proceedings of the 15th international conference on Parallel architectures and compilation techniques PACT '06**
Publisher: ACM Press

Full text available: pdf(159.54 KB) Additional Information: full citation, abstract, references, index terms

Modern compiler transformations that eliminate redundant computations or reorder instructions, such as partial redundancy elimination and instruction scheduling, are very effective in improving application performance but tend to create longer and potentially more complex live ranges. Typically the task of dealing with the increased register pressure is left to the register allocator. To avoid introduction of spill code which can reduce or completely eliminate the benefit of earlier optimization ...

Keywords: Itanium, VLIW, register allocation, register pressure, rematerialization

6 EPIC compilation: Optimization for the Intel® Itanium® architecture register stack

Alex Settle, Daniel A. Connors, Gerolf Hoflehner, Dan Lavery
March 2003 **Proceedings of the international symposium on Code generation and optimization: feedback-directed and runtime optimization CGO '03**
Publisher: IEEE Computer Society

Full text available: pdf(906.60 KB) Additional Information: full citation, abstract, references, citings, index terms

The Intel® Itanium® architecture contains a number of innovative compiler-controllable features designed to exploit instruction level parallelism. New code generation and optimization techniques are critical to the application of these features to improve processor performance. For instance, the Itanium® architecture provides a compiler-controllable virtual register stack to reduce the penalty of memory accesses associated with procedure calls. The Itanium® Register Stack Engine ...

7 Compiler scheduling: Reduced code size modulo scheduling in the absence of hardware support

Josep Llosa, Stefan M. Freudenberger

November 2002 **Proceedings of the 35th annual ACM/IEEE international symposium on Microarchitecture MICRO 35**

Publisher: IEEE Computer Society Press

Full text available: pdf(1.17 MB) Additional Information: full citation, abstract, references, index terms

Publisher Site

Modulo scheduling is a very effective instruction scheduling technique that exploits Instruction Level Parallelism (ILP) in loop bodies by overlapping the execution of successive iterations. Unfortunately, modulo scheduling has been shown to cause heavy code expansion. To avoid the penalties of code expansion, some processors have dedicated hardware support for modulo scheduled loops. However, this dedicated hardware support has a cost in chip area, cycle time, processor complexity, and compiler ...

8 Register allocation for software pipelined loops

B. R. Rau, M. Lee, P. P. Tirumalai, M. S. Schlansker

July 1992 **ACM SIGPLAN Notices , Proceedings of the ACM SIGPLAN 1992 conference on Programming language design and implementation PLDI '92**, Volume 27 Issue 7

Publisher: ACM Press

Full text available: pdf(1.84 MB) Additional Information: full citation, abstract, references, citings, index terms

Software pipelining is an important instruction scheduling technique for efficiently overlapping successive iterations of loops and executing them in parallel. This paper studies the task of register allocation for software pipelined loops, both with and without hardware features that are specifically aimed at supporting software pipelines. Register allocation for software pipelines presents certain novel problems leading to unconventional solutions, especially in the presence of hardware s ...

9 Modulo scheduling: Modulo schedule buffers

Matthew C. Merten, Wen-mei W. Hwu

December 2001 **Proceedings of the 34th annual ACM/IEEE international symposium on Microarchitecture MICRO 34**

Publisher: IEEE Computer Society

Full text available: pdf(1.38 MB) Additional Information: full citation, abstract, references, citings

Publisher Site

As VLIW/EPIC processors are increasingly used in real-time, signal-processing, and embedded applications, the importance of minimizing code size and reducing power is growing. This paper describes a new architectural mechanism, called the Modulo Schedule Buffers, that provides an elegant interface for the execution of modulo scheduled loops. While the performance is similar to that of kernel-only modulo scheduling, this mechanism has a number of advantages, including minimal code expansion. Rath ...

10 Smalltalk-80: the language and its implementation

Adele Goldberg, David Robson

January 1983 Book

Publisher: Addison-Wesley Longman Publishing Co., Inc.

Full text available: pdf(33.56 MB) Additional Information: full citation, abstract, cited by, index terms, review

From the Preface (See Front Matter for full Preface)

Advances in the design and production of computer hardware have brought many more people into direct contact with computers. Similar advances in the design and production of computer software are required in order that this increased contact be as rewarding as

possible. The Smalltalk-80 system is a result of a decade of research into creating computer software that is appropriate for producing highly functional and interactive ...

11 An open-source CVE for programming education: a case study: An open-source CVE for programming education: a case study

Andrew M. Phelps, Christopher A. Egert, Kevin J. Bierre, David M. Parks
July 2005 **ACM SIGGRAPH 2005 Courses SIGGRAPH '05**
Publisher: ACM Press
Full text available: pdf(7.92 MB) Additional Information: full citation, references

12 Software pipelining

Vicki H. Allan, Reese B. Jones, Randall M. Lee, Stephen J. Allan
September 1995 **ACM Computing Surveys (CSUR)**, Volume 27 Issue 3
Publisher: ACM Press
Full text available: pdf(4.72 MB) Additional Information: full citation, abstract, references, citings, index terms

Utilizing parallelism at the instruction level is an important way to improve performance. Because the time spent in loop execution dominates total execution time, a large body of optimizations focuses on decreasing the time to execute each iteration. Software pipelining is a technique that reforms the loop so that a faster execution rate is realized. Iterations are executed in overlapped fashion to increase parallelism.Let {ABC}n
Keywords: instruction level parallelism, loop reconstruction, optimization, software pipelining

13 A programming language

Kenneth E. Iverson
January 1962 Book
Publisher: John Wiley & Sons, Inc.
Additional Information: full citation, abstract, references, cited by, index terms

From the Preface

Applied mathematics is largely concerned with the design and analysis of explicit procedures for calculating the exact or approximate values of various functions. Such explicit procedures are called algorithms or programs. Because an effective notation for the description of programs exhibits considerable syntactic structure, it is called a programming language.

Much of applied mathematics, particularly the more recent computer-related areas which ...

14 Accelerator: using data parallelism to program GPUs for general-purpose uses

David Tarditi, Sidd Puri, Jose Oglesby
October 2006 **ACM SIGARCH Computer Architecture News , ACM SIGOPS Operating Systems Review , ACM SIGPLAN Notices , Proceedings of the 12th international conference on Architectural support for programming languages and operating systems ASPLOS-XII**, Volume 34 , 40 , 41 Issue 5 , 5 , 11
Publisher: ACM Press
Full text available: pdf(266.52 KB) Additional Information: full citation, abstract, references, index terms

GPUs are difficult to program for general-purpose uses. Programmers can either learn graphics APIs and convert their applications to use graphics pipeline operations or they can use stream programming abstractions of GPUs. We describe Accelerator, a system that uses data parallelism to program GPUs for general-purpose uses instead. Programmers use a conventional imperative programming language and a library that

provides only high-level data-parallel operations. No aspects of GPUs are exposed to ...

Keywords: data parallelism, graphics processing units, just-in time compilation

15 Artificial intelligence

Elaine Rich
January 1983 Book
Publisher: McGraw-Hill, Inc.
Additional Information: full citation, abstract, references, cited by, review

The goal of this book is to provide programmers and computer scientists with a readable introduction to the problems and techniques of artificial intelligence (A.I.). The book can be used either as a text for a course on A.I. or as a self-study guide for computer professionals who want to learn what A.I. is all about.

The book was designed as the text for a one-semester, introductory graduate course in A.I. In such a course, it should be possible to cover all of the material in the boo ...

16 Register allocation for software pipelined multi-dimensional loops

Hongbo Rong, Alban Douillet, Guang R. Gao
June 2005 **ACM SIGPLAN Notices , Proceedings of the 2005 ACM SIGPLAN conference on Programming language design and implementation PLDI '05**, Volume 40 Issue 6
Publisher: ACM Press
Full text available: pdf(745.17 KB) Additional Information: full citation, abstract, references, index terms

Software pipelining of a multi-dimensional loop is an important optimization that overlaps the execution of successive outermost loop iterations to explore instruction-level parallelism from the entire n-dimensional iteration space. This paper investigates register allocation for software pipelined multi-dimensional loops.For single loop software pipelining, the lifetime instances of a loop variant in successive iterations of the loop form a repetitive pattern. An effective register allocation m ...

Keywords: register allocation, software pipelining

17 Single-dimension software pipelining for multidimensional loops

Hongbo Rong, Zhizhong Tang, R. Govindarajan, Alban Douillet, Guang R. Gao
March 2007 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 4 Issue 1
Publisher: ACM Press
Full text available: pdf(439.64 KB) Additional Information: full citation, abstract, references, index terms

Traditionally, software pipelining is applied either to the innermost loop of a given loop nest or from the innermost loop to outer loops. This paper proposes a three-step approach, called *single-dimension software pipelining (SSP)*, to software pipeline a loop nest at an arbitrary loop level that has a rectangular iteration space and contains no sibling inner loops in it. The first step identifies the most profitable loop level for software pipelining in terms of initiation rate, data ...

Keywords: Software pipelining, loop transformation, modulo scheduling

18 Real-time shading

Marc Olano, Kurt Akeley, John C. Hart, Wolfgang Heidrich, Michael McCool, Jason L. Mitchell, Randi Rost
August 2004 **ACM SIGGRAPH 2004 Course Notes SIGGRAPH '04**
Publisher: ACM Press
Full text available: pdf(7.39 MB) Additional Information: full citation, abstract

Real-time procedural shading was once seen as a distant dream. When the first version of this course was offered four years ago, real-time shading was possible, but only with one-of-a-kind hardware or by combining the effects of tens to hundreds of rendering passes. Today, almost every new computer comes with graphics hardware capable of interactively executing shaders of thousands to tens of thousands of instructions. This course has been redesigned to address today's real-time shading capabili ...

19 Optimal register reassignment for register stack overflow minimization

Yoonseo Choi, Hwansoo Han

March 2006 **ACM Transactions on Architecture and Code Optimization (TACO)**, Volume 3 Issue 1

Publisher: ACM Press

Full text available: pdf(584.90 KB) Additional Information: full citation, abstract, references, index terms

Architectures with a register stack can implement efficient calling conventions. Using the overlapping of callers' and callees' registers, callers are able to pass parameters to callees without a memory stack. The most recent instance of a register stack can be found in the Intel Itanium architecture. A hardware component called the register stack engine (RSE) provides an illusion of an infinite-length register stack using a memory-backed process to handle overflow and underflow for a physically ...

Keywords: Register assignment, register allocation, register stack, sequence graph

20 Measurement and evaluation of the MIPS architecture and processor

Thomas R. Gross, John L. Hennessy, Steven A. Przybylski, Christopher Rowen

August 1988 **ACM Transactions on Computer Systems (TOCS)**, Volume 6 Issue 3

Publisher: ACM Press

Full text available: pdf(2.30 MB) Additional Information: full citation, abstract, references, citings, index terms, review

MIPS is a 32-bit processor architecture that has been implemented as an nMOS VLSI chip. The instruction set architecture is RISC-based. Close coupling with compilers and efficient use of the instruction set by compiled programs were goals of the architecture. The MIPS architecture requires that the software implement some constraints in the design that are normally considered part of the hardware implementation. This paper presents experimental results on the effectiveness of this processor ...

The ACM Portal is published by the Association for Computing Machinery. Copyright © 2007 ACM, Inc.
Terms of Usage Privacy Policy Code of Ethics Contact Us

Useful downloads: Adobe Acrobat QuickTime Windows Media Player Real Player

Results for "((register and allocat*)<in>metadata)"
Your search matched **502** of **1621473** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

e-mail printer friendly

» Search Options

View Session History

New Search

Modify Search

((register and allocat*)<in>metadata) Search

☐ Check to search only within this results set

Display Format: ◉ Citation ○ Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IET JNL	IET Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IET CNF	IET Conference Proceeding
IEEE STD	IEEE Standard

view selected items Select All Deselect All View: **1-25** | 26-50 | 51-75 | 76-100

1. **Precise register allocation for irregular architectures**
Kong, T.; Wilken, K.D.;
Microarchitecture, 1998. MICRO-31. Proceedings. 31st Annual ACM/IEEE International Symposium on
30 Nov.-2 Dec. 1998 Page(s):297 - 307
Digital Object Identifier 10.1109/MICRO.1998.742791
AbstractPlus | Full Text: PDF(344 KB) **IEEE CNF**
Rights and Permissions

2. **Late allocation and early release of physical registers**
Monreal, T.; Vinals, V.; Gonzalez, J.; Gonzalez, A.; Valero, M.;
Computers, IEEE Transactions on
Volume 53, Issue 10, Oct. 2004 Page(s):1244 - 1259
Digital Object Identifier 10.1109/TC.2004.79
AbstractPlus | References | Full Text: PDF(1696 KB) **IEEE JNL**
Rights and Permissions

3. **A progressive register allocator for irregular architectures**
Koes, D.; Goldstein, S.C.;
Code Generation and Optimization, 2005. CGO 2005. International Symposium on
20-23 March 2005 Page(s):269 - 279
Digital Object Identifier 10.1109/CGO.2005.4
AbstractPlus | Full Text: PDF(272 KB) **IEEE CNF**
Rights and Permissions

4. **Resolving register bank conflicts for a network processor**
Zhuang, X.; Santosh Pande;
Parallel Architectures and Compilation Techniques, 2003. PACT 2003. Proceedings. 12th International Conference on
27 Sept.-1 Oct. 2003 Page(s):269 - 278
Digital Object Identifier 10.1109/PACT.2003.1238022
AbstractPlus | Full Text: PDF(284 KB) **IEEE CNF**
Rights and Permissions

5. **Quantitative evaluation of the register stack engine and optimizations for future Itanium processors**
Weldon, R.D.; Chang, S.S.; Hong Wang; Hoflehner, G.; Wang, P.H.; Lavery, D.; Shen, J.P.;
Interaction between Compilers and Computer Architectures, 2002. Proceedings. Sixth Annual Workshop on
3 Feb. 2002 Page(s):57 - 67
Digital Object Identifier 10.1109/INTERA.2002.995843
AbstractPlus | Full Text: PDF(417 KB) **IEEE CNF**

Rights and Permissions

6. **Register organization for enhanced on-chip parallelism**
Sangireddy, R.;
Application-Specific Systems, Architectures and Processors, 2004. Proceedings. 15th IEEE International Conference on
2004 Page(s):180 - 190
Digital Object Identifier 10.1109/ASAP.2004.1342469
AbstractPlus | Full Text: PDF(387 KB) IEEE CNF
Rights and Permissions

7. **Delaying physical register allocation through virtual-physical registers**
Monreal, T.; Gonzalez, A.; Valero, M.; Gonzalez, J.; Vinals, V.;
Microarchitecture, 1999. MICRO-32. Proceedings. 32nd Annual International Symposium on
16-18 Nov. 1999 Page(s):186 - 192
Digital Object Identifier 10.1109/MICRO.1999.809456
AbstractPlus | Full Text: PDF(48 KB) IEEE CNF
Rights and Permissions

8. **Register write specialization register read specialization: a path to complexity-effective wide-issue superscalar processors**
Seznec, A.; Toullec, E.; Rochecouste, O.;
Microarchitecture, 2002. (MICRO-35). Proceedings. 35th Annual IEEE/ACM International Symposium on
18-22 Nov. 2002 Page(s):383 - 394
Digital Object Identifier 10.1109/MICRO.2002.1176265
AbstractPlus | Full Text: PDF(294 KB) IEEE CNF
Rights and Permissions

9. **Virtual registers**
Gonzalez, A.; Valero, M.; Gonzalez, J.; Monreal, T.;
High Performance Computing, 1997. Proceedings. Fourth International Conference on
18-21 Dec. 1997 Page(s):364 - 369
Digital Object Identifier 10.1109/HIPC.1997.634516
AbstractPlus | Full Text: PDF(572 KB) IEEE CNF
Rights and Permissions

10. **Virtual-physical registers**
Gonzalez, A.; Gonzalez, J.; Valero, M.;
High-Performance Computer Architecture, 1998. Proceedings., 1998 Fourth International Symposium on
1-4 Feb. 1998 Page(s):175 - 184
Digital Object Identifier 10.1109/HPCA.1998.650557
AbstractPlus | Full Text: PDF(44 KB) IEEE CNF
Rights and Permissions

11. **Customization of Register File Banking Architecture for Low Power**
Nalluri, R.; Garg, R.; Panda, P.R.;
VLSI Design, 2007. Held jointly with 6th International Conference on Embedded Systems., 20th International Conference on
6-10 Jan. 2007 Page(s):239 - 244
Digital Object Identifier 10.1109/VLSID.2007.58
AbstractPlus | Full Text: PDF(204 KB) IEEE CNF
Rights and Permissions

12. **Software-directed register deallocation for simultaneous multithreaded processors**
Lo, J.L.; Parekh, S.S.; Eggers, S.J.; Levy, H.M.; Tullsen, D.M.;
Parallel and Distributed Systems, IEEE Transactions on
Volume 10, Issue 9, Sept. 1999 Page(s):922 - 933
Digital Object Identifier 10.1109/71.798316
AbstractPlus | References | Full Text: PDF(1096 KB) IEEE JNL
Rights and Permissions

13. **Performance evaluation of register allocator for the advanced DSP of TMS320C80**
Jihong Kim; Short, G.;
Acoustics, Speech, and Signal Processing, 1998. ICASSP '98. Proceedings of the 1998 IEEE International Conference on
Volume 5, 12-15 May 1998 Page(s):3077 - 3080 vol.5
Digital Object Identifier 10.1109/ICASSP.1998.678176
AbstractPlus | Full Text: PDF(416 KB) **IEEE CNF**
Rights and Permissions

14. **Register allocation for design of data format converters**
Parhi, K.K.; Lee, J.-S.;
Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on
14-17 April 1991 Page(s):1133 - 1136 vol.2
Digital Object Identifier 10.1109/ICASSP.1991.150567
AbstractPlus | Full Text: PDF(552 KB) **IEEE CNF**
Rights and Permissions

15. **Storage allocation strategies for data path synthesis of ASICs**
Ramakrishna, N.A.; Bayoumi, M.A.;
Circuits and Systems, 1994. ISCAS '94., 1994 IEEE International Symposium on
Volume 1, 30 May-2 June 1994 Page(s):41 - 44 vol.1
Digital Object Identifier 10.1109/ISCAS.1994.408750
AbstractPlus | Full Text: PDF(328 KB) **IEEE CNF**
Rights and Permissions

16. **Register Pointer Architecture for Efficient Embedded Processors**
JongSoo Park; Sung-Boem Park; Balfour, J.D.; Black-Schaffer, D.; Kozyrakis, C.; Dally, W.J.;
Design, Automation & Test in Europe Conference & Exhibition, 2007. DATE '07
16-20 April 2007 Page(s):1 - 6
Digital Object Identifier 10.1109/DATE.2007.364659
AbstractPlus | Full Text: PDF(654 KB) **IEEE CNF**
Rights and Permissions

17. **Tailoring graph-coloring register allocation for runtime compilation**
Cooper, K.D.; Dasgupta, A.;
Code Generation and Optimization, 2006. CGO 2006. International Symposium on
26-29 March 2006 Page(s):11 pp.
Digital Object Identifier 10.1109/CGO.2006.35
AbstractPlus | Full Text: PDF(440 KB) **IEEE CNF**
Rights and Permissions

18. **A global, dynamic register allocation and binding for a data path synthesis system**
Woo, N.-S.;
Design Automation Conference, 1990. Proceedings. 27th ACM/IEEE
24-28 June 1990 Page(s):505 - 510
Digital Object Identifier 10.1109/DAC.1990.114908
AbstractPlus | Full Text: PDF(476 KB) **IEEE CNF**
Rights and Permissions

19. **Efficient Register Mapping and Allocation in LaTTe, an Open-Source Java Just-in-Time Compiler**
Byung-Sun Yang; Junpyo Lee; Seungil Lee; Seongbae Park; Chung, Y.C.; Suhyun Kim; Ebcioglu, K.; Altman, E.; Erik Altman;
Parallel and Distributed Systems, IEEE Transactions on
Volume 18, Issue 1, Jan. 2007 Page(s):57 - 69
Digital Object Identifier 10.1109/TPDS.2007.253281
AbstractPlus | Full Text: PDF(4139 KB) **IEEE JNL**
Rights and Permissions

20. **Exploiting quiescent states in register lifetime**
Sangireddy, R.; Somani, A.K.;

Computer Design: VLSI in Computers and Processors, 2004. ICCD 2004. Proceedings. IEEE International Conference on
11-13 Oct. 2004 Page(s):368 - 374
Digital Object Identifier 10.1109/ICCD.2004.1347948

AbstractPlus | Full Text: PDF(373 KB) IEEE CNF
Rights and Permissions

21. **Low-cost register-pressure prediction for scalar replacement using pseudo-schedules**
Ma, Y.; Carr, S.; Ge, R.;
Parallel Processing, 2004. ICPP 2004. International Conference on
2004 Page(s):116 - 124 vol.1
Digital Object Identifier 10.1109/ICPP.2004.1327911

AbstractPlus | Full Text: PDF(299 KB) IEEE CNF
Rights and Permissions

22. **Reducing register ports for higher speed and lower energy**
Park, I.; Powell, M.D.; Vijaykumar, T.N.;
Microarchitecture, 2002. (MICRO-35). Proceedings. 35th Annual IEEE/ACM International Symposium on
18-22 Nov. 2002 Page(s):171 - 182
Digital Object Identifier 10.1109/MICRO.2002.1176248

AbstractPlus | Full Text: PDF(277 KB) IEEE CNF
Rights and Permissions

23. **A faster optimal register allocator**
Changqing Fu; Wilken, K.;
Microarchitecture, 2002. (MICRO-35). Proceedings. 35th Annual IEEE/ACM International Symposium on
18-22 Nov. 2002 Page(s):245 - 256
Digital Object Identifier 10.1109/MICRO.2002.1176254

AbstractPlus | Full Text: PDF(562 KB) IEEE CNF
Rights and Permissions

24. **A novel renaming scheme to exploit value temporal locality through physical register reuse and unification**
Jourdan, S.; Ronen, R.; Bekerman, M.; Shomar, B.; Yoaz, A.;
Microarchitecture, 1998. MICRO-31. Proceedings. 31st Annual ACM/IEEE International Symposium on
30 Nov.-2 Dec. 1998 Page(s):216 - 225
Digital Object Identifier 10.1109/MICRO.1998.742783

AbstractPlus | Full Text: PDF(128 KB) IEEE CNF
Rights and Permissions

25. **A graph-theoretic approach for register file based synthesis**
Ravikumar, C.P.; Aggarwal, R.; Sharma, C.;
VLSI Design, 1997. Proceedings., Tenth International Conference on
4-7 Jan. 1997 Page(s):118 - 123
Digital Object Identifier 10.1109/ICVD.1997.567972

AbstractPlus | Full Text: PDF(500 KB) IEEE CNF
Rights and Permissions

View: **1-25** | 26-50 | 51-75 | 76-100

Indexed by Inspec

© Copyright 2006 IEEE – All Rights Reserved

Results for "**((rotat* and register and schedul*)<in>metadata)**"
Your search matched **10** of **1621473** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

e-mail printer friendly

» Search Options

View Session History

New Search

Modify Search

((rotat* and register and schedul*)<in>metadata) Search

Check to search only within this results set

Display Format: Citation Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IET JNL	IET Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IET CNF	IET Conference Proceeding
IEEE STD	IEEE Standard

view selected items Select All Deselect All

1. **A technique for throughput and register optimization during resource constrained pipelined scheduling**
Rangan, N.; Chatha, K.S.;
VLSI Design, 2005. 18th International Conference on
2005 Page(s):564 - 569
Digital Object Identifier 10.1109/ICVD.2005.35
AbstractPlus | Full Text: PDF(208 KB) **IEEE CNF**
Rights and Permissions

2. **RCRS: a framework for loop scheduling with limited number of registers**
Kaisheng Wang; Ted Zhihong; Yu Edwin; Sha, H.-M.;
VLSI, 1998. Proceedings of the 8th Great Lakes Symposium on
19-21 Feb. 1998 Page(s):386 - 391
Digital Object Identifier 10.1109/GLSV.1998.665328
AbstractPlus | Full Text: PDF(200 KB) **IEEE CNF**
Rights and Permissions

3. **Loop scheduling algorithm for timing and memory operation minimization with register constraint**
Chen, F.; Tongsima, S.; Sha, E.H.-M.;
Signal Processing Systems, 1998. SIPS 98. 1998 IEEE Workshop on
8-10 Oct. 1998 Page(s):579 - 588
Digital Object Identifier 10.1109/SIPS.1998.715820
AbstractPlus | Full Text: PDF(296 KB) **IEEE CNF**
Rights and Permissions

4. **Register queues: a new hardware/software approach to efficient software pipelining**
Smelyanskiy, M.; Tyson, G.S.; Davidson, E.S.;
Parallel Architectures and Compilation Techniques, 2000. Proceedings. International Conference on
15-19 Oct. 2000 Page(s):3 - 12
Digital Object Identifier 10.1109/PACT.2000.888255
AbstractPlus | Full Text: PDF(940 KB) **IEEE CNF**
Rights and Permissions

5. **Evaluating the use of register queues in software pipelined loops**
Tyson, G.S.; Smelyanskiy, M.; Davidson, E.S.;
Computers, IEEE Transactions on
Volume 50, Issue 8, Aug. 2001 Page(s):769 - 783
Digital Object Identifier 10.1109/12.946998
AbstractPlus | References | Full Text: PDF(3240 KB) **IEEE JNL**
Rights and Permissions

6. **Code generation for single-dimension software pipelining of multi-dimensional loops**
Rong, H.; Douillet, A.; Govindarajan, R.; Gao, G.R.;
Code Generation and Optimization, 2004. CGO 2004. International Symposium on
2004 Page(s):175 - 186
Digital Object Identifier 10.1109/CGO.2004.1281673
AbstractPlus | Full Text: PDF(713 KB) IEEE CNF
Rights and Permissions

7. **The Intel IA-64 compiler code generator**
Bharadwaj, J.; Chen, W.Y.; Chuang, W.; Hoflehner, G.; Menezes, K.; Muthukumar, K.; Pierce, J.;
Micro, IEEE
Volume 20, Issue 5, Sept.-Oct. 2000 Page(s):44 - 53
Digital Object Identifier 10.1109/40.877949
AbstractPlus | References | Full Text: PDF(164 KB) IEEE JNL
Rights and Permissions

8. **Applying data speculation in modulo scheduled loops**
Mahadevan, U.; Nomura, K.; Ju, R.D.-C.; Hank, R.;
Parallel Architectures and Compilation Techniques, 2000. Proceedings. International Conference on
15-19 Oct. 2000 Page(s):169 - 176
Digital Object Identifier 10.1109/PACT.2000.888341
AbstractPlus | Full Text: PDF(660 KB) IEEE CNF
Rights and Permissions

9. **Constraint satisfaction for relative location assignment and scheduling**
Alba-Pinto, C.; Mesman, B.; Jess, J.;
Computer Aided Design, 2001. ICCAD 2001. IEEE/ACM International Conference on
4-8 Nov. 2001 Page(s):384 - 390
Digital Object Identifier 10.1109/ICCAD.2001.968652
AbstractPlus | Full Text: PDF(534 KB) IEEE CNF
Rights and Permissions

10. **Optimizing software data prefetches with rotating registers**
Doshi, G.; Krishnaiyer, R.; Muthukumar, K.;
Parallel Architectures and Compilation Techniques, 2001. Proceedings. 2001 International Conference on
8-12 Sept. 2001 Page(s):257 - 267
Digital Object Identifier 10.1109/PACT.2001.953306
AbstractPlus | Full Text: PDF(800 KB) IEEE CNF
Rights and Permissions

Indexed by Inspec

© Copyright 2006 IEEE – All Rights Reserved

Results for **"(spill* and rotat* and register<in>metadata)"**
Your search matched **31** of **1621473** documents.
A maximum of **100** results are displayed, **25** to a page, sorted by **Relevance** in **Descending** order.

» Search Options

View Session History

New Search

Modify Search

(spill* and rotat* and register<in>metadata) Search

☐ Check to search only within this results set

Display Format: ◉ Citation ○ Citation & Abstract

» Key

IEEE JNL	IEEE Journal or Magazine
IET JNL	IET Journal or Magazine
IEEE CNF	IEEE Conference Proceeding
IET CNF	IET Conference Proceeding
IEEE STD	IEEE Standard

view selected items Select All Deselect All 1-25 | 26-31

1. **Quantitative evaluation of the register stack engine and optimizations for future Itanium processors**
 Weldon, R.D.; Chang, S.S.; Hong Wang; Hoflehner, G.; Wang, P.H.; Lavery, D.; Shen, J.P.;
 Interaction between Compilers and Computer Architectures, 2002. Proceedings. Sixth Annual Workshop on
 3 Feb. 2002 Page(s):57 - 67
 Digital Object Identifier 10.1109/INTERA.2002.995843
 AbstractPlus | Full Text: PDF(417 KB) IEEE CNF
 Rights and Permissions

2. **Evaluating the use of register queues in software pipelined loops**
 Tyson, G.S.; Smelyanskiy, M.; Davidson, E.S.;
 Computers, IEEE Transactions on
 Volume 50, Issue 8, Aug. 2001 Page(s):769 - 783
 Digital Object Identifier 10.1109/12.946998
 AbstractPlus | References | Full Text: PDF(3240 KB) IEEE JNL
 Rights and Permissions

3. **Optimization for the Intel/spl reg/ Itanium/spl reg/ architecture register stack**
 Settle, A.; Connors, D.A.; Hoflehner, G.; Lavery, D.;
 Code Generation and Optimization, 2003. CGO 2003. International Symposium on
 23-26 March 2003 Page(s):115 - 124
 Digital Object Identifier 10.1109/CGO.2003.1191538
 AbstractPlus | Full Text: PDF(285 KB) IEEE CNF
 Rights and Permissions

4. **Module scheduling with reduced register pressure**
 Llosa, J.; Valero, M.; Agyuade, E.; Gonzalez, A.;
 Computers, IEEE Transactions on
 Volume 47, Issue 6, June 1998 Page(s):625 - 638
 Digital Object Identifier 10.1109/12.689643
 AbstractPlus | References | Full Text: PDF(604 KB) IEEE JNL
 Rights and Permissions

5. **Register constrained modulo scheduling**
 Zalamea, J.; Llosa, J.; Ayguade, E.; Valero, M.;
 Parallel and Distributed Systems, IEEE Transactions on
 Volume 15, Issue 5, May 2004 Page(s):417 - 430
 Digital Object Identifier 10.1109/TPDS.2004.1278099
 AbstractPlus | References | Full Text: PDF(1454 KB) IEEE JNL
 Rights and Permissions

6. **Register aware scheduling for distributed cache clustered architecture**
Zhong Wang; Xiaobo Sharon Hu; Sha, E.H.-M.;
Design Automation Conference, 2003. Proceedings of the ASP-DAC 2003. Asia and South Pacific
21-24 Jan. 2003 Page(s):71 - 76
AbstractPlus | Full Text: PDF(764 KB) **IEEE CNF**
Rights and Permissions

7. **RS-FDRA: A register-sensitive software pipelining algorithm for embedded VLIW processors**
Akturan, C.; Jacome, M.F.;
Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on
Volume 21, Issue 12, Dec. 2002 Page(s):1395 - 1415
Digital Object Identifier 10.1109/TCAD.2002.804373
AbstractPlus | References | Full Text: PDF(1081 KB) **IEEE JNL**
Rights and Permissions

8. **The Intel IA-64 compiler code generator**
Bharadwaj, J.; Chen, W.Y.; Chuang, W.; Hoflehner, G.; Menezes, K.; Muthukumar, K.; Pierce, J.;
Micro, IEEE
Volume 20, Issue 5, Sept.-Oct. 2000 Page(s):44 - 53
Digital Object Identifier 10.1109/40.877949
AbstractPlus | References | Full Text: PDF(164 KB) **IEEE JNL**
Rights and Permissions

9. **Phi-predication for light-weight if-conversion**
Weihaw Chuang; Calder, B.; Ferrante, J.;
Code Generation and Optimization, 2003. CGO 2003. International Symposium on
23-26 March 2003 Page(s):179 - 190
Digital Object Identifier 10.1109/CGO.2003.1191544
AbstractPlus | Full Text: PDF(386 KB) **IEEE CNF**
Rights and Permissions

10. **Overview of the FREEDOM compiler for mapping DSP software to FPGAs**
Zaretsky, D.; Mittal, M.; Xiaoyong Tang; Banerjee, P.;
Field-Programmable Custom Computing Machines, 2004. FCCM 2004. 12th Annual IEEE Symposium on
20-23 April 2004 Page(s):37 - 46
Digital Object Identifier 10.1109/FCCM.2004.44
AbstractPlus | Full Text: PDF(400 KB) **IEEE CNF**
Rights and Permissions

11. **Adapting EPIC Architecture’s Register Stack for Virtual Stack Machines**
Tayeb, J.; Niar, S.;
Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006. 9th EUROMICRO Conference on
0-0 0 Page(s):204 - 210
Digital Object Identifier 10.1109/DSD.2006.20
AbstractPlus | Full Text: PDF(292 KB) **IEEE CNF**
Rights and Permissions

12. **Stream register files with indexed access**
Jayasena, N.; Erez, M.; Ahn, J.H.; Dally, W.J.;
High Performance Computer Architecture, 2004. HPCA-10. Proceedings. 10th International Symposium on
14-18 Feb. 2004 Page(s):60 - 72
Digital Object Identifier 10.1109/HPCA.2004.10007
AbstractPlus | Full Text: PDF(432 KB) **IEEE CNF**
Rights and Permissions

13. **Unroll-based copy elimination for enhanced pipeline scheduling**

Suhyun Kim; Soo-Mook Moon; Jinpyo Park; Ebcioglu, K.;
Computers, IEEE Transactions on
Volume 51, Issue 9, Sept. 2002 Page(s):977 - 994
Digital Object Identifier 10.1109/TC.2002.1032620

AbstractPlus | References | Full Text: PDF(5232 KB) IEEE JNL
Rights and Permissions

14. **Exploring the design space of an optimized compiler approach for mesh-like coarse-grained reconfigurable architectures**
Dimitroulakos, G.; Galanis, M.D.; Goutis, C.E.;
Parallel and Distributed Processing Symposium, 2006. IPDPS 2006. 20th International
25-29 April 2006 Page(s):10 pp.
Digital Object Identifier 10.1109/IPDPS.2006.1639349

AbstractPlus | Full Text: PDF(256 KB) IEEE CNF
Rights and Permissions

15. **Lifetime-sensitive modulo scheduling in a production environment**
Llosa, J.; Ayguade, E.; Gonzalez, A.; Valero, M.; Eckhardt, J.;
Computers, IEEE Transactions on
Volume 50, Issue 3, March 2001 Page(s):234 - 249
Digital Object Identifier 10.1109/12.910814

AbstractPlus | References | Full Text: PDF(1276 KB) IEEE JNL
Rights and Permissions

16. **A methodology for specifying PET VOIs using multimodality techniques**
Klein, G.J.; Teng, X.; Jagust, W.J.; Eberling, J.L.; Acharya, A.; Reutter, B.W.; Huesman, R.H.;
Medical Imaging, IEEE Transactions on
Volume 16, Issue 4, Aug. 1997 Page(s):405 - 415
Digital Object Identifier 10.1109/42.611350

AbstractPlus | References | Full Text: PDF(232 KB) IEEE JNL
Rights and Permissions

17. **Absolute quantitation of myocardial activity in phantoms**
Da Silva, A.J.; Tang, H.R.; Wu, M.C.; Hasegawa, B.H.;
Nuclear Science, IEEE Transactions on
Volume 46, Issue 3, Part 2, June 1999 Page(s):659 - 666
Digital Object Identifier 10.1109/23.775595

AbstractPlus | References | Full Text: PDF(1104 KB) IEEE JNL
Rights and Permissions

18. **Resource constrained modulo scheduling for coarse-grained reconfigurable arrays**
Dimitroulakos, G.; Galanis, M.D.; Goutis, C.E.;
Circuits and Systems, 2006. ISCAS 2006. Proceedings. 2006 IEEE International Symposium on
21-24 May 2006 Page(s):4 pp.
Digital Object Identifier 10.1109/ISCAS.2006.1693231

AbstractPlus | Full Text: PDF(4296 KB) IEEE CNF
Rights and Permissions

19. **Resource constrained dataflow retiming heuristics for VLIW ASIPs**
Jacome, M.; de Veciana, G.; Akturan, C.;
Hardware/Software Codesign, 1999. (CODES '99) Proceedings of the Seventh International Workshop on
3-5 May 1999 Page(s):12 - 16
Digital Object Identifier 10.1109/HSC.1999.777383

AbstractPlus | Full Text: PDF(424 KB) IEEE CNF
Rights and Permissions

20. **Evaluating the use of register queues in software pipelined loops**
Tyson, G.S.; Smelyanskiy, M.; Davidson, E.S.;
Computers, IEEE Transactions on
Volume 50, Issue 8, Aug. 2001 Page(s):769 - 783
Digital Object Identifier 10.1109/TC.2001.947006

AbstractPlus | Full Text: PDF(3240 KB) IEEE JNL

Rights and Permissions

21. **Review of current proposed technologies for mass storage systems**
Matick, R.E.;
Proceedings of the IEEE
Volume 60, Issue 3, March 1972 Page(s):266 - 289
AbstractPlus | Full Text: PDF(3014 KB) IEEE JNL
Rights and Permissions

22. **Ozone monitoring instrument calibration**
Dobber, M.R.; Dirksen, R.J.; Levelt, P.F.; van den Oord, G.H.J.; Voors, R.H.M.; Kleipool, Q.; Jaross, G.; Kowalewski, M.; Hilsenrath, E.; Leppelmeier, G.W.; Johan de Vries; Dierssen, W.; Rozemeijer, N.C.;
Geoscience and Remote Sensing, IEEE Transactions on
Volume 44, Issue 5, May 2006 Page(s):1209 - 1238
Digital Object Identifier 10.1109/TGRS.2006.869987
AbstractPlus | Full Text: PDF(3912 KB) IEEE JNL
Rights and Permissions

23. **Study of relative quantification of Tc-99 m with partial volume effect and spillover correction for SPECT oncology imaging**
Guido Boening; Pretorius, P.H.; King, M.A.;
Nuclear Science, IEEE Transactions on
Volume 53, Issue 3, Part 2, June 2006 Page(s):1205 - 1212
Digital Object Identifier 10.1109/TNS.2006.871406
AbstractPlus | Full Text: PDF(1656 KB) IEEE JNL
Rights and Permissions

24. **High-speed coherence processing using the sectionalized Fourier transform**
Gerlach, A.;
Acoustics, Speech, and Signal Processing [see also IEEE Transactions on Signal Processing], IEEE Transactions on
Volume 30, Issue 2, Apr 1982 Page(s):189 - 205
AbstractPlus | Full Text: PDF(1640 KB) IEEE JNL
Rights and Permissions

25. **Magnetic film domain-wall motion devices**
Spain, R.; Marino, M.;
Magnetics, IEEE Transactions on
Volume 6, Issue 3, Sep 1970 Page(s):451 - 463
AbstractPlus | Full Text: PDF(1856 KB) IEEE JNL
Rights and Permissions

1-25 | 26-31

Indexed by Inspec®

© Copyright 2006 IEEE – All Rights Reserved

Web

Results **1** - **10** of about **397** for **rotating register allocation compil***. (**0.45** seconds)

Method, apparatus, and product for optimizing compiler with ...
Modulo scheduling and **rotating register allocation**/assignment introduce additional ... to the time required for the compiler to **compile a** source program. ...
www.patentstorm.us/patents/6651247-description.html - 141k - Supplemental Result - Cached - Similar pages

[PDF] Differential **Register Allocation**
File Format: PDF/Adobe Acrobat - View as HTML
register allocation. We demonstrate that differential **register** alloca- ... aged **rotating** registers (for example in the Itanium processor) could ...
www.cc.gatech.edu/~xt2000/papers/pldi2005.pdf - Supplemental Result - Similar pages

System and method for improved **register allocation** in an ...
The improved optimizing compiler coordinates **register allocation** and assignment by **rotating** and scalar **register** allocators to generate efficient global (ie, ...
www.freepatentsonline.com/20030237080.html - 65k - Supplemental Result - Cached - Similar pages

[PS] **Register Allocation** for Software Pipelined Multi-dimensional Loops
File Format: Adobe PostScript - View as Text
Due to the cyclic nature of the **rotating register** file, the **register** ... Unknown length of an interval at **compile time**. A variant ...
www.capsl.udel.edu/pub/doc/papers/SSP-RegAlloc.ps.gz - Supplemental Result - Similar pages

[PS] Region-based **Register Allocation** for EPIC Architectures
File Format: Adobe PostScript - View as Text
Modulo scheduling and **Rotating register**. allocations 8.4 Comparisons of **compile time** and execution time for dierent region ...
www.cs.nyu.edu/web/Research/Theses/hansoo.ps.gz - Supplemental Result - Similar pages

[PPT] **Register Allocation**
File Format: Microsoft Powerpoint - View as HTML
Thus **compile time** is comparable to Chow-Hennessy ... **Register Allocation**. For loop variants only. If a **rotating register** is allocated to physical **register** r ...
cs.nyu.edu/courses/spring01/G22.3033-005/**register_allocation**_palem_wong.ppt - Supplemental Result - Similar pages

[PDF] f Early Periodic **Register Allocation** on ILP Processors 1. Introduction
File Format: PDF/Adobe Acrobat - View as HTML
register allocation is described in Sect. 4 and formulated with integer linear programming. (intLP). The special case where a **rotating register** file (RRF) ...
www.prism.uvsq.fr/~touati/publis/PPL-03-037.pdf - Supplemental Result - Similar pages

System, method, and apparatus for spilling and filling **rotating** ...
The spilling and filling of non-**rotating** registers reduces the live ranges and **register** pressure thus enabling **register allocation**. ...
www.patentdebate.com/PATAPP/20050071607 - Supplemental Result - Similar pages

[PS] SIRA: Schedule Independent **Register Allocation** for Software Pipelining
File Format: Adobe PostScript - View as Text
The **register allocation** in loops is generally carried out af- ... **compile time**. Our approach is to take into account the. registers constraints before the ...

www.icsa.informatics.ed.ac.uk/cpc2001/Proceedings/eisenbeis.ps - Supplemental Result - Similar pages

[PDF] Early Control of **Register** Pressure for Software Pipelined Loops
File Format: PDF/Adobe Acrobat
Hence, the problem of cyclic **register allocation** with minimal critical circuit on **rotating**. **register** files can be stated as follows. Problem 2 (SIRA HAM). ...
www.springerlink.com/index/7H5QFLRRAJN4PLCX.pdf - Supplemental Result - Similar pages

1 2 3 4 5 6 7 8 9 10 **Next**

Download Google Pack: free essential software for your PC

rotating register allocation compil* Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

©2007 Google - Google Home - Advertising Programs - Business Solutions - About Google

Web

Results **1** - **10** of about **540** for **rotating register allocation schedul***. (**0.26** seconds)

[PS] SIRA: **Schedule Independent Register Allocation** for Software Pipelining
File Format: Adobe PostScript - View as Text
cause doing the **register allocation** at first step without as-. suming a **schedule lacks** the information of interferences. between values live ranges. ...
www.icsa.informatics.ed.ac.uk/cpc2001/Proceedings/eisenbeis.ps - Supplemental Result - Similar pages

[PDF] SIRA: **Schedule Independent Register Allocation** for Software Pipelining
File Format: PDF/Adobe Acrobat - View as HTML
SIRA: **Schedule Independent Register Allocation** for Software. Pipelining. Sid Ahmed Ali Touati, ... **rotating register** files [ELM95, RLTS92]. This however ...
www.prism.uvsq.fr/~touati/publis/cpc01-touati.pdf - Supplemental Result - Similar pages

Method, apparatus, and product for optimizing compiler with ...
In a computer having **rotating** registers, a **schedule**-assigner for allocating the ... Modulo scheduling and **rotating register allocation**/assignment introduce ...
www.patentstorm.us/patents/6651247-description.html - 141k - Supplemental Result - Cached - Similar pages

Method, apparatus, and product for optimizing compiler with ...
The **scheduler**-assigner of claim 5, wherein: when the second **rotating register** is ... Modulo scheduling and **rotating register allocation**/assignment introduce ...
www.patentmonkey.com/PM/patentid/6651247.aspx - 244k - Supplemental Result - Cached - Similar pages

Method, apparatus, and product for optimizing compiler with ...
The **scheduler**-assigner also includes a **rotating register** allocator that ... Modulo scheduling and **rotating register allocation**/assignment introduce ...
www.freepatentsonline.com/6651247.html - 149k - Supplemental Result - Cached - Similar pages

System and method for improved **register allocation** in an ...
A method for improving **register allocation** in an optimizing compiler, ... The modulo **schedule instruction** generator 650 constructs a **rotating register** IR ...
www.freepatentsonline.com/20030237080.html - 65k - Supplemental Result - Cached - Similar pages

[PDF] Differential **Register Allocation**
File Format: PDF/Adobe Acrobat - View as HTML
register allocation. We demonstrate that differential **register** alloca- ... aged **rotating** registers (for example in the Itanium processor) could ...
www.cc.gatech.edu/~xt2000/papers/pldi2005.pdf - Supplemental Result - Similar pages

[PS] **Register Allocation** for Software Pipelined Multi-dimensional Loops
File Format: Adobe PostScript - View as Text
Register Allocation For Software Pipelined. Loop Nests. This section briefly reviews SSP, characterizes the vector lifetimes. in an SSP **schedule**, ...
www.capsl.udel.edu/pub/doc/papers/SSP-RegAlloc.ps.gz - Supplemental Result - Similar pages

[PS] Region-based **Register Allocation** for EPIC Architectures
File Format: Adobe PostScript - View as Text
(II) The **register allocation** phase can be sensitive. to the other compiler optimization phases. For instance, if the **scheduler optimizes** ...

www.cs.nyu.edu/web/Research/Theses/hansoo.ps.gz - Supplemental Result - Similar pages

[PDF] A New Fast Algorithm for Optimal **Register Allocation** in Modulo ...
File Format: PDF/Adobe Acrobat - View as HTML
schedule with a maximum computation throughput of a loop ie minimize the initiation ...
2.1.1 **Rotating Register** File. We consider the **allocation** of variable ...
ftp://ftp.inria.fr/INRIA/publication/publi-pdf/RR/RR-3337.pdf - Supplemental Result - Similar pages

1 2 3 4 5 6 7 8 9 10 **Next**

Try Google Desktop: search your computer as easily as you search the web.

rotating register allocation schedul* Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

©2007 Google - Google Home - Advertising Programs - Business Solutions - About Google

Web

Results **1 - 10** of about **7,290** for **rotating register spill* fill*** . (**0.33** seconds)

Method, apparatus, and product for optimizing compiler with ...
In some aspects, step (a) includes, when a **rotating register** is not available for having a live range designated to it, (i), inserting one or more **spill** ...
www.patentstorm.us/patents/6651247-description.html - 141k - Supplemental Result - Cached - Similar pages

Methods and apparatus for efficient control of floating-point ...
The **rotating register** permits multiple iterations of the program loop to be executed in ... In other processor operations, known as **spill/fill operations**, ...
www.patentstorm.us/patents/6151669-description.html - 69k - Supplemental Result - Cached - Similar pages

Method, apparatus, and product for optimizing compiler with ...
(i) when the second **rotating register** is not available, inserting one or more **spill** code instructions in the first IR for the second live range, and ...
www.freepatentsonline.com/6651247.html - 149k - Supplemental Result - Cached - Similar pages

Method and apparatus to reduce **spill and fill overhead** in a ...
However, the invention may be practiced in other kinds of processors that may wish to use selective **spill and fill of register** contents. ...
www.freepatentsonline.com/20050138340.html - 53k - Supplemental Result - Cached - Similar pages
[More results from www.freepatentsonline.com]

[PDF] Performance Advantage of the **Register** Stack in Intel® Itanium ...
File Format: PDF/Adobe Acrobat - View as HTML
stacked registers may be used as **rotating** registers during ... **Spill**. **Fill**. Figure 4 Snapshot of the Itanium **Register** Stack in Operation. ...
systems.cs.colorado.edu/EPIC2/papers/s2-1-rakvic.pdf - Supplemental Result - Similar pages

[PDF] Prematerialization: Reducing **Register** Pressure for Free
File Format: PDF/Adobe Acrobat - View as HTML
to **rotate** in the context of a software pipelined loop. The registers ... processor is stalled waiting for a **register** window to **spill/fill can** ...
www.cs.virginia.edu/~pact2006/program/pact2006/pact146-baev2.pdf - Supplemental Result - Similar pages

Spill containment pan
Compressible gasket members are used to **fill the** gap between the side walls and the ... Transmission interlock system Clarifier with **rotating** sealed cover ...
www.patentopedia.us/motor_with_electronic_distributing_configuration/**spill**_containment_pan.html - 53k - Supplemental Result - Cached - Similar pages

[PDF] Optimal **Register** Reassignment for **Register** Stack Overflow Minimization
File Format: PDF/Adobe Acrobat
a significant amount of **register spill/fill overhead** involved in the **register** stack ... the procedures that use **rotating**-registers from our optimization. ...
portal.acm.org/ft_gateway.cfm?id=1132467&type=pdf - Supplemental Result - Similar pages

[PDF] Inter-Procedural Stacked **Register** Allocation for Itanium Like ...

File Format: PDF/Adobe Acrobat
stacked registers decreases **spill** overhead (**spill** variables to. memory) at the expense of potential increases of **register** stack. overflow and **fill. ...**
portal.acm.org/ft_gateway.cfm?id=782844&type=pdf&dl=GUIDE&dl=ACM - Supplemental Result - Similar pages

EPA: Federal **Register**: Oil and Gas Extraction Point Source ...
Spill Prevention, Control, and Countermeasure D. ... These **fill materials** require a **fill permit** issued pursuant to section 404 of the CWA. ...
www.epa.gov/EPA-WATER/1996/December/Day-16/pr-21096DIR/pr-21096.html - 250k - Supplemental Result - Cached - Similar pages

1 2 3 4 5 6 7 8 9 10 **Next**

Try Google Desktop: search your computer as easily as you search the web.

rotating register spill* fill* Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

©2007 Google - Google Home - Advertising Programs - Business Solutions - About Google

Web

Results **1 - 10** of about **213,000** for **rotating register spill* load***. (**0.21** seconds)

[PDF] Microsoft PowerPoint - 2_robert
File Format: PDF/Adobe Acrobat - View as HTML
Rotating registers are spilled. FP **register spill** is usually outside loop. 20. Spilling ALAT Slots. • Data speculation: compiler moves a **load** ...
rogue.colorado.edu/EPIC4/presentations/2_robert_6pclr.pdf - Similar pages

[PDF] Itanium
File Format: PDF/Adobe Acrobat - View as HTML
stack frame and **rotating register** base for GRs. The CFM holds the ... perform **register spill** and reload operations in the background when neces- ...
www.cse.unsw.edu.au/~cs9244/06/seminars/07-gaol.pdf - Similar pages

[PDF] Itanium Outline History Application **Register** State
File Format: PDF/Adobe Acrobat - View as HTML
Renaming **register** number = **rotate register** number +. value of rrb. **Register** Rotation (cont) ... Invalidating ALAT entries: by advanced **load,** certain ...
www.cse.unsw.edu.au/~cs9244/06/seminars/07-gaol-t.pdf - Similar pages

[PDF] Improved **Spill** Code Generation for Software Pipelined Loops
File Format: PDF/Adobe Acrobat
tence of a **rotating register** le and use the wands{only strat- ... This **spill** requires. the use of several **load and** store operations and may ...
portal.acm.org/ft_gateway.cfm?id=349319&type=pdf&dl=portal&dl=ACM - Similar pages

[PDF] Heuristics for **Register**-constrained Software Pipelining *
File Format: PDF/Adobe Acrobat
cates the addition of **spill load**/store operations without. affecting the whole schedule. A **rotating register** file can be used to solve this ...
portal.acm.org/ft_gateway.cfm?id=243892&type=pdf&dl=portal&dl=ACM - Similar pages
[More results from portal.acm.org]

Method, apparatus, and product for optimizing compiler with ...
In some aspects, step (a) includes, when a **rotating register** is not available for having a live range designated to it, (i), inserting one or more **spill** ...
www.patentstorm.us/patents/6651247-description.html - 141k - Cached - Similar pages

[PDF] **Register** constrained modulo scheduling - Parallel and Distributed ...
File Format: PDF/Adobe Acrobat
A **rotating register**. file can be used to solve this problem without replicating this, the Early_Start of a **spill load node** is set to its Late_Start ...
ieeexplore.ieee.org/iel5/71/28575/01278099.pdf?arnumber=1278099 - Similar pages

System and method for improved **register** allocation in an ...
The **register** spiller 640 uses the **rotating register** usage information 634 and the interfering lifetimes 532 to determine when to **spill** the contents of ...
www.freepatentsonline.com/20030237080.html - 65k - Cached - Similar pages

[PDF] The CPU IA-64: Key Features to Improve Performance
File Format: PDF/Adobe Acrobat - View as HTML
addition to that, the object **register, load address** and number of bytes being will **spill** the old state registers to the **register** backing store memory. ...
gec.di.uminho.pt/discip/minf/ac0304/ICCA04/Proc/21200_IA64.pdf - Similar pages

[PDF] Quantitative Evaluation of **Register** Pressure on Software Pipelined ...

File Format: PDF/Adobe Acrobat
loop-variant variables (**rotating register** file), all the current superscalar ... indirectly, since **spill** code adds new **load and** store operations, ...
www.springerlink.com/index/R688860281177357.pdf - Similar pages

1 2 3 4 5 6 7 8 9 10 **Next**

Download Google Pack: free essential software for your PC

rotating register spill* load* Search

Search within results | Language Tools | Search Tips | Dissatisfied? Help us improve

©2007 Google - Google Home - Advertising Programs - Business Solutions - About Google